



---

## RTC overview



---

## Contents

---

1. RTC overview .....	3
2. How to use the RTC .....	10
3. Menuconfig or how to configure kernel .....	17
4. RTC device tree configuration .....	24



A quality version of this page, approved on *14 September 2021*, was based off this revision.

This article gives information about the Linux<sup>®</sup> RTC framework. The RTC framework is involved in precise time countdown.

## Contents

1 Framework purpose .....	4
2 System overview .....	5
2.1 Component description .....	5
2.2 API description .....	5
3 Configuration .....	6
3.1 Kernel Configuration .....	6
3.2 Device tree configuration .....	6
4 How to use the framework .....	7
5 How to trace and debug the framework .....	8
5.1 How to trace .....	8
6 Source code location .....	9
7 References .....	10



---

## 1 Framework purpose

---

The RTC keeps the system time up to date and can wake up the system from the standby power mode at a programmed time.

A general presentation of the RTC framework is available in the Linux RTC documentation <sup>[1]</sup>.



---

## 2 System overview

---

### RTC framework overview

#### 2.1 Component description

- **RTC (hardware)**  
RTC dedicated hardware block in STM32MPU product.
- **rtc-stm32**  
RTC ST driver.
- **rtc**  
Linux RTC framework, it provides API to RTC driver and interfaces to user.
- **Sysfs interface**  
Sysfs interface is accessible via `/sys/class/rtc/rtcX/`.
- **Procfs interface**  
Procfs interface is accessible via `/proc/driver/rtc`.
- **Char device interface**  
Device interface is accessible via `/dev/rtcX`.
- **User application**  
The user application can be an user built application or an community application like hwclock.

#### 2.2 API description

API is described in Linux RTC documentation <sup>[1]</sup>.



---

## 3 Configuration

---

### 3.1 Kernel Configuration

Activate **rtc-stm32** driver in kernel configuration using the Linux Menuconfig tool: Menuconfig or how to configure kernel

```
Device drivers --->
  *- Real Time Clock --->
    <*> STM32 RTC
```

### 3.2 Device tree configuration

Please refer to the [RTC device tree configuration](#).



---

## 4 How to use the framework

---

Please refer to the How to use the RTC.



---

## 5 How to trace and debug the framework

---

### 5.1 How to trace

Dynamic debug traces can be added using the following commands:

```
echo -n 'file rtc-stm32.c +p'>/sys/kernel/debug/dynamic_debug/control
```





---

## 6 Source code location

---

- rtc-stm32: drivers/rtc/rtc-stm32.c
- api: include/linux/rtc.h
- framework:
  - drivers/rtc/class.c
  - drivers/rtc/dev.c
  - drivers/rtc/interface.c
  - drivers/rtc/lib.c
  - drivers/rtc/proc.c
  - drivers/rtc/sysfs.c
- device-tree bindings constants: include/dt-bindings/rtc/rtc-stm32.h



## 7 References

- 1.01.1 Linux RTC documentation

Stable: 09.09.2021 - 14:00 / Revision: 07.09.2021 - 14:49

This article gives information about the Linux<sup>®</sup> RTC framework. The RTC framework is involved in precise time countdown.

### Contents

1 Framework purpose .....	11
2 System overview .....	12
2.1 Component description .....	12
2.2 API description .....	12
3 Configuration .....	13
3.1 Kernel Configuration .....	13
3.2 Device tree configuration .....	13
4 How to use the framework .....	14
5 How to trace and debug the framework .....	15
5.1 How to trace .....	15
6 Source code location .....	16
7 References .....	17



---

## 1 Framework purpose

---

The RTC keeps the system time up to date and can wake up the system from the standby power mode at a programmed time.

A general presentation of the RTC framework is available in the Linux RTC documentation <sup>[1]</sup>.



---

## 2 System overview

---

### RTC framework overview

#### 2.1 Component description

- **RTC (hardware)**  
RTC dedicated hardware block in STM32MPU product.
- **rtc-stm32**  
RTC ST driver.
- **rtc**  
Linux RTC framework, it provides API to RTC driver and interfaces to user.
- **Sysfs interface**  
Sysfs interface is accessible via `/sys/class/rtc/rtcX/`.
- **Procfs interface**  
Procfs interface is accessible via `/proc/driver/rtc`.
- **Char device interface**  
Device interface is accessible via `/dev/rtcX`.
- **User application**  
The user application can be an user built application or an community application like hwclock.

#### 2.2 API description

API is described in Linux RTC documentation <sup>[1]</sup>.



---

## 3 Configuration

---

### 3.1 Kernel Configuration

Activate **rtc-stm32** driver in kernel configuration using the Linux Menuconfig tool: Menuconfig or how to configure kernel

```
Device drivers --->
  *- Real Time Clock --->
    <*> STM32 RTC
```

### 3.2 Device tree configuration

Please refer to the [RTC device tree configuration](#).



---

## 4 How to use the framework

---

Please refer to the [How to use the RTC](#).



---

## 5 How to trace and debug the framework

---

### 5.1 How to trace

Dynamic debug traces can be added using the following commands:

```
echo -n 'file rtc-stm32.c +p'>/sys/kernel/debug/dynamic_debug/control
```



---

## 6 Source code location

---

- rtc-stm32: drivers/rtc/rtc-stm32.c
- api: include/linux/rtc.h
- framework:
  - drivers/rtc/class.c
  - drivers/rtc/dev.c
  - drivers/rtc/interface.c
  - drivers/rtc/lib.c
  - drivers/rtc/proc.c
  - drivers/rtc/sysfs.c
- device-tree bindings constants: include/dt-bindings/rtc/rtc-stm32.h





## 7 References

- 1.01.1 Linux RTC documentation

Stable: 31.03.2021 - 08:47 / Revision: 26.03.2021 - 08:44

This article gives information about the Linux<sup>®</sup> RTC framework. The RTC framework is involved in precise time countdown.

### Contents

1 Framework purpose .....	18
2 System overview .....	19
2.1 Component description .....	19
2.2 API description .....	19
3 Configuration .....	20
3.1 Kernel Configuration .....	20
3.2 Device tree configuration .....	20
4 How to use the framework .....	21
5 How to trace and debug the framework .....	22
5.1 How to trace .....	22
6 Source code location .....	23
7 References .....	24



---

## 1 Framework purpose

---

The RTC keeps the system time up to date and can wake up the system from the standby power mode at a programmed time.

A general presentation of the RTC framework is available in the Linux RTC documentation <sup>[1]</sup>.



---

## 2 System overview

---

### RTC framework overview

#### 2.1 Component description

- **RTC (hardware)**  
RTC dedicated hardware block in STM32MPU product.
- **rtc-stm32**  
RTC ST driver.
- **rtc**  
Linux RTC framework, it provides API to RTC driver and interfaces to user.
- **Sysfs interface**  
Sysfs interface is accessible via `/sys/class/rtc/rtcX/`.
- **Procfs interface**  
Procfs interface is accessible via `/proc/driver/rtc`.
- **Char device interface**  
Device interface is accessible via `/dev/rtcX`.
- **User application**  
The user application can be an user built application or an community application like `hwclock`.

#### 2.2 API description

API is described in Linux RTC documentation <sup>[1]</sup>.



---

## 3 Configuration

---

### 3.1 Kernel Configuration

Activate **rtc-stm32** driver in kernel configuration using the Linux Menuconfig tool: Menuconfig or how to configure kernel

```
Device drivers --->
  *- Real Time Clock --->
    <*> STM32 RTC
```

### 3.2 Device tree configuration

Please refer to the [RTC device tree configuration](#).



---

## 4 How to use the framework

---

Please refer to the [How to use the RTC](#).



---

## 5 How to trace and debug the framework

---

### 5.1 How to trace

Dynamic debug traces can be added using the following commands:

```
echo -n 'file rtc-stm32.c +p'>/sys/kernel/debug/dynamic_debug/control
```



---

## 6 Source code location

---

- rtc-stm32: drivers/rtc/rtc-stm32.c
- api: include/linux/rtc.h
- framework:
  - drivers/rtc/class.c
  - drivers/rtc/dev.c
  - drivers/rtc/interface.c
  - drivers/rtc/lib.c
  - drivers/rtc/proc.c
  - drivers/rtc/sysfs.c
- device-tree bindings constants: include/dt-bindings/rtc/rtc-stm32.h



## 7 References

- 1.01.1 Linux RTC documentation

Stable: 09.09.2021 - 14:07 / Revision: 09.09.2021 - 14:07

This article gives information about the Linux<sup>®</sup> RTC framework. The RTC framework is involved in precise time countdown.

### Contents

1 Framework purpose .....	25
2 System overview .....	26
2.1 Component description .....	26
2.2 API description .....	26
3 Configuration .....	27
3.1 Kernel Configuration .....	27
3.2 Device tree configuration .....	27
4 How to use the framework .....	28
5 How to trace and debug the framework .....	29
5.1 How to trace .....	29
6 Source code location .....	30
7 References .....	31





---

## 1 Framework purpose

---

The RTC keeps the system time up to date and can wake up the system from the standby power mode at a programmed time.

A general presentation of the RTC framework is available in the Linux RTC documentation <sup>[1]</sup>.



---

## 2 System overview

---

### RTC framework overview

#### 2.1 Component description

- **RTC (hardware)**  
RTC dedicated hardware block in STM32MPU product.
- **rtc-stm32**  
RTC ST driver.
- **rtc**  
Linux RTC framework, it provides API to RTC driver and interfaces to user.
- **Sysfs interface**  
Sysfs interface is accessible via `/sys/class/rtc/rtcX/`.
- **Procfs interface**  
Procfs interface is accessible via `/proc/driver/rtc`.
- **Char device interface**  
Device interface is accessible via `/dev/rtcX`.
- **User application**  
The user application can be an user built application or an community application like `hwclock`.

#### 2.2 API description

API is described in Linux RTC documentation <sup>[1]</sup>.



---

## 3 Configuration

---

### 3.1 Kernel Configuration

Activate **rtc-stm32** driver in kernel configuration using the Linux Menuconfig tool: Menuconfig or how to configure kernel

```
Device drivers --->
  *- Real Time Clock --->
    <*> STM32 RTC
```

### 3.2 Device tree configuration

Please refer to the [RTC device tree configuration](#).



---

## 4 How to use the framework

---

Please refer to the [How to use the RTC](#).



---

## 5 How to trace and debug the framework

---

### 5.1 How to trace

Dynamic debug traces can be added using the following commands:

```
echo -n 'file rtc-stm32.c +p'>/sys/kernel/debug/dynamic_debug/control
```



---

## 6 Source code location

---

- rtc-stm32: drivers/rtc/rtc-stm32.c
- api: include/linux/rtc.h
- framework:
  - drivers/rtc/class.c
  - drivers/rtc/dev.c
  - drivers/rtc/interface.c
  - drivers/rtc/lib.c
  - drivers/rtc/proc.c
  - drivers/rtc/sysfs.c
- device-tree bindings constants: include/dt-bindings/rtc/rtc-stm32.h



---

## 7 References

---

- 1.01.1 Linux RTC documentation